Io

a small programming language
Purpose

briefly show Io’s Lua roots

present overview of Io

get your feedback

working together
Some history

interested in dynamic OO languages since 1990

did NeXTstep/ObjC and Python development

found Lua - a great language

used Lua on Yindo project
A new language

liked Lua’s size and speed but...

willing to trade off for greater simplicity

wanted a pure OO language
Lua and Io

- small
- simple
- highly dynamic
- multi-platform
- multi-state
- BSD/MIT licensed
- designed for embedding
- incremental garbage collection
- syntax that script writers can deal with
Lua

faster

smaller

more mature

larger community
io

pure OO

no globals

code is data

lazily evaluated arguments

simpler, more consistent syntax and semantics
Io overview

**simple**  prototype-based object model
   all actions are messages
   simple and consistent syntax

**dynamic**  all messages are dynamic
   code is data and runtime modifiable

**concurrent**  all objects can be actors
   actors use coroutines
   futures supported

**and...**  bundled with extensive official bindings
The language

no keywords
no statements (only expressions)
expressions are composed only of messages
supports lexically scoped blocks
objects can have multiple parents
## Message Syntax

<table>
<thead>
<tr>
<th>Lua</th>
<th>Io</th>
</tr>
</thead>
<tbody>
<tr>
<td>a:b()</td>
<td>a b</td>
</tr>
<tr>
<td>a:b(c)</td>
<td>a b(c)</td>
</tr>
<tr>
<td>a:b(c, d)</td>
<td>a b(c, d)</td>
</tr>
</tbody>
</table>
# Operators

<table>
<thead>
<tr>
<th>expression</th>
<th>compiles to</th>
</tr>
</thead>
<tbody>
<tr>
<td>a * 2 * b</td>
<td>a *(2) *(b)</td>
</tr>
</tbody>
</table>
## Assignment

<table>
<thead>
<tr>
<th>expression</th>
<th>compiles to</th>
</tr>
</thead>
<tbody>
<tr>
<td><code>a := 2</code></td>
<td><code>setSlot(&quot;a&quot;, 2)</code></td>
</tr>
<tr>
<td><code>a = 2</code></td>
<td><code>updateSlot(&quot;a&quot;, 2)</code></td>
</tr>
</tbody>
</table>

This separation allows self to be implicit
Loops

while(x < 10, ...)  
for(i, 1, 10, ...)  
loop(...)  
10 repeatTimes(...)

Conditions

a := if(b == 1, c, d) // conditions are expressions

if(a == b) then(
    ...
)
elseif(...) then(
    ...
)
)
someList := list("a", 2.3, "foo")
someList foreach(i, v, writeln(i, " : ", v))

// foreach also works on Maps, Strings, Buffers, etc
Blocks and Methods

foo := method(a, a + b)  // object scoped

foo := block(a, a + b)  // lexically scoped
Scoping

no globals

variables are local by default
Expressions

a := people select(person, person age < 30)

names := people map(i, person, person name)
"Macro" Example

glChunk := method(
    glPushMatrix
    sender doMessage(thisMessage argAt(0))
    glPopMatrix
)

glChunk(glTranslated(1,2,3); glRectd(0,0,100,100))
Objects

Account := Object clone do(
    balance := 0
    deposit := method(amount,
        balance = balance + amount
    )
)
)
Example

account := Account clone
account deposit(10.00)
writeln("balance:", account balance)
Number double := method(self * 2)

100 double

==> 200
Introspection

Number double := method(self * 2)
Number getSlot(“double”) code
==> “method(self *(2))”
url := URL with("http://www.google.com")

<table>
<thead>
<tr>
<th>url fetch</th>
<th>// sync message</th>
</tr>
</thead>
<tbody>
<tr>
<td>f := url @fetch</td>
<td>// future message</td>
</tr>
<tr>
<td>url @@fetch</td>
<td>// async message</td>
</tr>
</tbody>
</table>

Futures auto-detect deadlocks
IoVM

Date (high precision, supports dates < 1970)
Duration
List
ImmutableSequence (Strings/Symbols)
Sequence (Buffers)
Map
WeakLink
IoServer

SGMLParser (supports XML and HTML)
Socket (async, libevent, supports async DNS)
Transparent Distributed Objects
Vector (supports SIMD/altivec)
Regex
SQLite3
MD5
Blowfish
CGI, URL
IoDesktop

OpenGL, GLU, GLUT
Audio (PortAudio)
Font (FreeType, caches in texture)
Movie (ffmpeg)
Ion user interface toolkit
Ion example
Implementation
Garbage Collector

non-moving, tri-color, write-barrier, generational
Tricks

objects use perfect hashes

lookups done by symbol

objects create hashes on demand

objects are recycled

block contexts are recycled immediately
Platforms

Unix     OSX, Linux, *BSD, Irix
Windows  Cygwin, Mingw, MSVC
Other    Symbian, Syllable, Zeta
What’s next?

Io 1.0 by end of 2005

incremental orthogonal persistence

packages

docs for Ion

bug tracker

revision control

official wiki
Working Together

bindings

Vector, Image, Movie, Font...
I’m interested to hear your thoughts and suggestions

steve@dekorte.com

more info at

iolanguage.com